EXTENSIBLE OBJECT PREVIEWER IN A SHELL BROWSER

TECHNICAL FIELD

The present invention relates generally to the field of computer software. More particularly, the invention relates to a system and method for extending the functionality of an object previewer in the shell or file system browser to improve the user experience.

BACKGROUND OF THE INVENTION

The need to readily identify items stored in a computing environment such as a personal computer (PC) is dramatically increasing as more individuals utilize computers in their daily routines and as the type of stored information varies between pictures, music, documents, etc. Documents and media are typically stored on computers in a hierarchical fashion and are organized with files of information or media stored within folders. File system browsers enable users to navigate through the file system and locate and open files and folders. For example, Microsoft Corporation's WINDOWS® EXPLORER™ is an operating system utility which enables users to browse the file system. However, many users find it difficult to correctly identify a file based on the information currently available in conventional file system browsers.

The ability to view metadata about the various files can greatly assist a user in finding a particular file stored on their computer. In the Microsoft Corporation's WINDOWS® 9X operating systems, a user can view object metadata by accessing the property sheet for a particular object. A property sheet presents the user with a list of the attributes or settings of an object in the form of a tabbed, index-card-like selection of property pages, each of which features standard dialog-style controls for customizing parameters. However, using the property sheet to locate items can be slow and cumbersome, and some users find it difficult to locate the relevant metadata in a property sheet. Similarly, the use of infotips to locate items can be slow and cumbersome because a user must hover the mouse cursor over each file in order to view the limited metadata displayed in an infotip.

Conventional file system browsers do not allow users to enter and edit metadata relating to files and folders, which would significantly enhance a user's ability to later locate that file. To date, the ability of users to enter and edit metadata has been limited to special purpose software programs. Examples of such programs include media players for electronic music files and application programs for electronic picture files. However, the use of such a program is limited to the particular type of file supported by the program, as opposed to a file system browser which supports multiple file types.

The Microsoft Corporation's WINDOWS® XP operating system includes an image browser for use in the My Pictures folder. The My Pictures folder is endowed with special features which enable users to view pictures as photos, not just as document icons. My Picture's image browsing features include the ability to view thumbnail-size and large versions of your photos, rotate photos that are flopped sideways, and create a slide show. You can also view a photo's details, such as its dimensions, the date and time it was taken, and the name of the camera that took it. The preview control area in the My Picture's folder contains an enlarged preview image of a user-selected image, iterator buttons to assist a user in iterating through a series of pictures and controls for rotating pictures in a clockwise or counterclockwise direction. These image browsing features are discussed in further detail below in connection with FIG. 2.

While the image browsing features discussed above have advanced the state of the art by alleviating the need to invoke a separate application program to view and manipulate pictures, the functionality provided by the preview control is limited to picture files and closed to third party software developers. There is little or no metadata displayed, and users cannot enter and edit metadata associated with the picture files. The preview image is a static viewer, as opposed to an interactive one. The ability to view only the first page of a multi-page document does not necessarily assist the user in recognizing file contents.

Accordingly, there is a need for a file system or shell browser which offers users improved file content recognition features so that users can readily locate their files. A need also exists for an improved graphical user interface for a shell browser which allows for the selection of a previewer for a

particular file type from a plurality of available previewers. There is also a need for an extensible shell browser which would allow software developers to provide additional information and functionality to users on a file type basis.

SUMMARY OF THE INVENTION

The present invention meets the above needs and overcomes one or more deficiencies in the prior art by providing a system and method for extending the functionality of an object previewer in a shell browser configured to display a plurality of items representing multiple item types, thereby improving the user experience.

In one aspect of the present invention, a shell browser is provided which includes a default previewer and an extensibility mechanism. The default previewer provides a standard level of functionality for multiple item types. The extensibility mechanism enables functionality beyond the standard level provided by the default previewer for one or more of the item types.

Another aspect of the present invention is a shell browser having a first previewer and a second previewer. The first previewer provides a standard level of functionality for multiple item types, and the second previewer provides an alternative or extended level of functionality for one or more of the multiple item types. The shell browser is configured to selectively deploy either the first previewer or the second previewer for the one or more item types.

In a further aspect, the present invention provides a graphical user interface for a shell browser which supports multiple item types. The graphical user interface includes a first screen area for displaying a set of items in the shell browser and means for selecting a previewer for the displayed items from a plurality of available previewers.

Still another aspect of the present invention is a computer-implemented method for selecting a previewer in a shell browser which supports multiple item types. The method includes providing a plurality of previewers in the shell browser for a particular item type and selecting one of the previewers for the particular item type. The method then associates the selected previewer with the particular item type.

In yet another aspect of the invention, a computer-implemented method is provided for enabling the use of third party previewers in a shell

browser which supports multiple item types. The method includes providing a shell browser having a default previewer for the multiple item types and providing an extensibility mechanism which enables a third party to develop an alternative previewer for at least one of the multiple item types.

A further aspect of the present invention is a data structure containing information indicative of a plurality of previewers in a shell browser. The data structure, which is stored on one or more computer-readable media, includes a first field containing information indicative of a default previewer which supports multiple item types. A second field contains information indicative of an alternative previewer for a first item type, and a third field contains information indicative of whether to invoke the default previewer or the alternative previewer when items of the first item type are displayed in the shell browser.

BRIEF DESCRIPTION OF THE DRAWING

The present invention is described in detail below with reference to the attached drawing figures, wherein:

FIG. 1 is a block diagram of a computing-system environment suitable for use in implementing the present invention;

FIG. 2 is a schematic diagram of a prior art graphical user interface for browsing pictures stored in a folder within a shell browser environment which is used for viewing other non-pictorial files and folders;

FIG. 3 is a block diagram of an exemplary graphical user interface for a shell browser;

FIG. 4 is a schematic diagram of a welcome pane in a shell browser;

FIG. 5 is a schematic diagram of a selected pane in a shell browser;

FIG. 6 is a schematic diagram of a selected pane in a shell browser with extended controls in accordance with an embodiment of the present invention;

FIG. 7 is a schematic diagram of a selected pane similar to FIG. 5 but including a context menu enabling a user to select a previewer in a shell browser in accordance with an embodiment of the present invention;

FIG. 8A is a flow diagram illustrating a method for enabling a user to select a previewer in a shell browser in accordance with an embodiment of the present invention;

FIG. 8B is a flow diagram illustrating a method for enabling the system to select a previewer in a shell browser in accordance with an embodiment of the present invention;

FIG. 9 is a flow diagram illustrating a method for enabling the use of third party previewers in a shell browser in accordance with an embodiment of the present invention; and

FIG. 10 is a block diagram of a data structure containing information indicative of multiple previewers in a shell browser.

DETAILED DESCRIPTION OF THE INVENTION

The present invention relates to a system and method for extending the functionality of an object previewer in a shell browser configured to display a plurality of items representing multiple item types, thereby improving the user experience. An exemplary operating environment for the present invention is described below.

Referring to the drawings in general and initially to FIG. 1 in particular, wherein like reference numerals identify like components in the various figures, an exemplary operating environment for implementing the present invention is shown and designated generally as operating environment 100. The computing system environment 100 is only one example of a suitable computing environment and is not intended to suggest any limitation as to the scope of use or functionality of the invention. Neither should the computing environment 100 be interpreted as having any dependency or requirement relating to any one or combination of components illustrated in the exemplary operating environment 100.

The invention may be described in the general context of computer-executable instructions, such as program modules, being executed by a computer. Generally, program modules include routines, programs, objects, components, data structures, etc., that perform particular tasks or implement particular abstract data types. Moreover, those skilled in the art will appreciate that the invention may be practiced with a variety of computer-system

configurations, including hand-held devices, multiprocessor systems, microprocessor-based or programmable-consumer electronics, minicomputers, mainframe computers, and the like. The invention may also be practiced in distributed-computing environments where tasks are performed by remote-processing devices that are linked through a communications network. In a distributed-computing environment, program modules may be located in both local and remote computer-storage media including memory storage devices.

With reference to FIG. 1, an exemplary system 100 for implementing the invention includes a general purpose computing device in the form of a computer 110 including a processing unit 120, a system memory 130, and a system bus 121 that couples various system components including the system memory 130 to the processing unit 120.

Computer 110 typically includes a variety of computer-readable media. By way of example, and not limitation, computer-readable media may comprise computer-storage media and communication media. Examples of computer-storage media include, but are not limited to, Random Access Memory (RAM); Read-Only Memory (ROM); Electronically Erasable Programmable Read-Only Memory (EEPROM); flash memory or other memory technology; CD-ROM, digital versatile discs (DVD) or other optical or holographic disc storage; magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices; or any other medium that can be used to store desired information and be accessed by computer 110. The system memory 130 includes computer-storage media in the form of volatile and/or nonvolatile memory such as ROM 131 and RAM 132. A Basic Input/Output System 133 (BIOS), containing the basic routines that help to transfer information between elements within computer 110 (such as during start-up) is typically stored in ROM 131. RAM 132 typically contains data and/or program modules that are immediately accessible to and/or presently being operated on by processing unit 120. By way of example, and not limitation, FIG. 1 illustrates operating system 134, application programs 135, other program modules 136, and program data 137.

The computer 110 may also include other removable/nonremovable, volatile/nonvolatile computer-storage media. By way of example only, FIG. 1 illustrates a hard disk drive 141 that reads from or writes

to nonremovable, nonvolatile magnetic media, a magnetic disk drive 151 that reads from or writes to a removable, nonvolatile magnetic disk 152, and an optical disc drive 155 that reads from or writes to a removable, nonvolatile optical disc 156 such as a CD-ROM or other optical media. Other removable/nonremovable, volatile/nonvolatile computer-storage media that can be used in the exemplary operating environment include, but are not limited to, magnetic tape cassettes, flash memory units, digital versatile disks, digital video tape, solid state RAM, solid state ROM, and the like. The hard disk drive 141 is typically connected to the system bus 121 through a nonremovable memory interface such as interface 140. Magnetic disk drive 151 and optical disc drive 155 are typically connected to the system bus 121 by a removable memory interface, such as interface 150.

The drives and their associated computer-storage media discussed above and illustrated in FIG. 1 provide storage of computer-readable instructions, data structures, program modules and other data for computer 110. For example, hard disk drive 141 is illustrated as storing operating system 144, application programs 145, other program modules 146, and program data 147. Note that these components can either be the same as or different from operating system 134, application programs 135, other program modules 136, and program data 137. Typically, the operating system, application programs and the like that are stored in RAM are portions of the corresponding systems, programs, or data read from hard disk drive 141, the portions varying in size and scope depending on the functions desired. Operating system 144, application programs 145, other program modules 146, and program data 147 are given different numbers here to illustrate that, at a minimum, they can be different copies. A user may enter commands and information into the computer 110 through input devices such as a keyboard 162; pointing device 161, commonly referred to as a mouse, trackball or touch pad; a wireless-input-reception component 163; or a wireless source such as a remote control. Other input devices (not shown) may include a microphone, joystick, game pad, satellite dish, scanner, or the like. These and other input devices are often connected to the processing unit 120 through a user-input interface 160 that is coupled to the system bus 121 but may be connected by

other interface and bus structures, such as a parallel port, game port, IEEE 1394 port, or a universal serial bus (USB), or infrared (IR) bus.

A display device 191 is also connected to the system bus 121 via an interface, such as a video interface 190. Display device 191 can be any device to display the output of computer 110 not limited to a monitor, an LCD screen, a Thin Film Transistor (TFT) screen, a flat-panel display, a conventional television, or screen projector. In addition to the display device 191, computers may also include other peripheral output devices such as speakers 197 and printer 196, which may be connected through an output peripheral interface 195.

The computer 110 in the present invention will operate in a networked environment using logical connections to one or more remote computers, such as a remote computer 180. The remote computer 180 may be a personal computer, and typically includes many or all of the elements described above relative to the computer 110, although only a memory storage device 181 has been illustrated in FIG. 1. The logical connections depicted in FIG. 1 include a local-area network (LAN) 171 and a wide-area network (WAN) 173 but may also include other networks, such as connections to a metropolitan-area network (MAN), intranet, or the Internet.

When used in a LAN networking environment, the computer 110 is connected to the LAN 171 through a network interface or adapter 170. When used in a WAN networking environment, the computer 110 typically includes a modem 172 or other means for establishing communications over the WAN 173, such as the Internet. The modem 172, which may be internal or external, may be connected to the system bus 121 via the network interface 170, or other appropriate mechanism. Modem 172 could be a cable modem, DSL modem, or other broadband device. In a networked environment, program modules depicted relative to the computer 110, or portions thereof, may be stored in the remote memory storage device. By way of example, and not limitation, FIG. 1 illustrates remote application programs 185 as residing on memory device 181. It will be appreciated that the network connections shown are exemplary and other means of establishing a communications link between the computers may be used.

Although many other internal components of the computer 110 are not shown, those of ordinary skill in the art will appreciate that such components

and the interconnections are well-known. For example, including various expansion cards such as television-tuner cards and network-interface cards within a computer 110 is conventional. Accordingly, additional details concerning the internal construction of the computer 110 need not be disclosed in connection with the present invention.

When the computer 110 is turned on or reset, the BIOS 133, which is stored in ROM 131, instructs the processing unit 120 to load the operating system, or necessary portion thereof, from the hard disk drive 141 into the RAM 132. Once the copied portion of the operating system, designated as operating system 144, is loaded into RAM 132, the processing unit 120 executes the operating-system code and causes the visual elements associated with the user interface of the operating system 134 to be displayed on the display device 191. Typically, when an application program 145 is opened by a user, the program code and relevant data are read from the hard disk drive 141 and the necessary portions are copied into RAM 132, the copied portion represented herein by reference numeral 135.

As previously mentioned, the present invention may be described in the general context of computer-useable instructions. Computer-useable instructions include functions, procedures, schemas, routines, code segments, and modules useable by one or more computers or other devices. The computer-useable instructions form an interface to allow a computer to react according to a source of input. The instructions cooperate with other code segments to initiate a variety of tasks in response to data received in conjunction with the source of the received data.

FIG. 2 is a schematic diagram of a prior art graphical user interface for browsing pictures stored in a folder within a shell browser environment which is used for viewing other non-pictorial files and folders. As stated above, the need to readily identify items that are stored in a computing environment such as a PC is dramatically increasing. With respect to digital pictures, users traditionally had to invoke a third party software program in order to view a specific file on the PC. FIG. 2 illustrates a prior solution, a film strip view, which allows users to more readily view and identify the image associated with a given file within the graphical operating environment. The goal of film

strip view was to alleviate the need for other software programs when browsing a folder of pictures by providing a quick iterative process that allows a user to preview a sizeable image of one or more picture files within the folder.

FIG. 2 relates to a system for browsing pictures stored in a folder, wherein a series of folder pictures is presented as a single row of thumbnails within an environment that is utilized for viewing other non-pictorial files and folders (i.e., a shell browser). It further allows a user to selectively cursor through the thumbnails, as it displays an enlarged preview image of a user selected thumbnail. FIG. 2 is a diagram of a representative window on a user's screen. As shown, the window 200 is divided into several areas including a header region, a task option area 206, a preview control area 202, a caption or comment area and a filmstrip area 204. The task option area 206 contains a list of tasks that can be selected by a user in order to perform a wide variety of operations relating to the management of files and folders, as well as other system choices. Some of these operations are specific to the pictures in the filmstrip area 204 and the preview control area 202. The preview control area 202 is a space in which an enlarged preview image of a user selected picture will be displayed. This space can also contain navigational icons to assist a user in iterating through a series of pictures. Immediately below the preview control area is a caption or comment area that can be utilized to display a variety of textual information. A film strip area 204, provides a space to display a single row of thumbnail images P1, P2, P3, P4 of the picture files contained within a given folder. In addition, the film strip area 204 also contains cursors to allow a user to scroll through a folder for the picture files. It should be noted that the filmstrip area 204 can contain and display thumbnail images in mixed orientation. For instance, as shown in FIG. 2, P1, P2 and P4 are in landscape while P3 is in portrait.

A user can select any one of the thumbnail images, which will cause a larger preview image of the user thumbnail selection image to be displayed within the preview control area. In addition, user selection of a thumbnail image will also allow the user to select and perform any one of the tasks listed in the task option area 206, with respect to the selected image. A first control button allows a user to quickly and successively preview an enlarged

image of each of the thumbnail images within a given folder, by iterating in one direction. In other words, a user would not have to specifically 'click' on each and every successive thumbnail image in order to preview the picture. Instead the user will merely click on the first control button repeatedly to move through the folder. A second control button performs a similar iteration function but only in the opposite direction.

Turning to FIG. 3, a window 300 represents a screen-size display area for a graphical user interface of a general purpose shell browser. The window 300 contains a preview pane area 302 and a view area 304. The preview pane 302 may include a preview control 306, an edit or metadata control 308, and a task control 310. Typically, the preview control 306 will provide the user with an image or other visual display of the item being previewed (e.g., a selected file). The preview control 306 may also present the user with controls such as iterator buttons which allow the user to shift the focus from one item to the next by clicking a mouse button. The edit control 308 not only includes a display of key properties of the previewed item, it also presents the user with a control for making edits to the metadata. The task control 310 contains tasks relevant to the namespace and/or the selection.

Those skilled in the art will appreciate that the present invention contemplates the presence of optional features within the window 300. For example, the metadata control 208 and the task control 210 are not essential features for purposes of the present invention. Moreover, other non-essential features which are not shown in FIG. 3, such as a toolbar which includes iterator buttons or a show/hide button so the user can open/close the preview pane, are also within the scope of the present invention. Nevertheless, these and other optional features may assist the user in readily locating a particular item in the shell browser.

The view area 304 provides a listview of one or more items 312, such as file system files or folders. The term "listview" refers to an enumeration or list of items within a container. The terms "item" and "shell item" are used interchangeably herein to refer to files, folders and other such containers, and other non-file objects which can be represented in a listview. Similarly, "shell item" refers to an item in a shell library. Examples of non-file objects may

include, but would not be limited to, contacts, favorites and email messages. The terms "shell browser" and "file system browser" are used interchangeably herein to refer to a browser which allows a user to navigate through various namespaces including files and other non-file items.

Those skilled in the art will appreciate that the present invention contemplates many possible designs and layouts for the window 300. For example, the preview pane 302 is shown above the view area 304 in FIG. 3. However, other layouts, such as placing the preview pane 302 and the view area 304 side-by-side, are clearly within the scope of the present invention. There are also many possible views for the items depicted in view area 304, such as details, slide show, filmstrip, thumbnail, tiles, icons, etc.

Referring next to FIG. 4, a schematic illustration is provided of a welcome pane 400 in a shell browser. A welcome pane is sometimes referred to as a "null select" pane because it represents a namespace or container as opposed to a selection. If the user has not yet made a selection, a preview pane 402 displays metadata 404 and key tasks relating to the folder or shell library. If desired, the tasks may be separated into premiered tasks 406 and other relevant tasks 408. The welcome pane 400 also includes a view area 410, in which multiple files or other items 412 may be viewed. The welcome pane metadata 404 may include information such as properties of the container (e.g., MyPictures), in which case the metadata display may be static. Alternatively, the welcome pane metadata 404 may include information such as a sampling of metadata from each of the items within the container, in which case the metadata display may change frequently. For example, the metadata display may be limited to properties of one item at a time by cycling from one item to the next every 30 seconds.

FIG. 5 is a schematic illustration of a selected pane 500 in a shell browser. As opposed to a welcome pane, a selected pane represents a selection by the user. If the user selects a container or folder, the selected pane need not be identical to the welcome pane for that container or folder. In FIG. 5, the selected pane 500 includes a preview pane 502 which contains a preview control 504, a metadata display 506 and a tasks display 508. Like the welcome pane 400 (in FIG. 4), the selected pane 500 also includes a view area 510, in which multiple

files or other items 512 may be viewed. In FIG. 5, however, the user has selected one of the files. Consequently, the preview control 504 displays a preview image of the selected file, the metadata display 506 shows properties of the selected file, and the tasks display 508 provides a menu of relevant tasks for operating on the selected file.

FIG. 6 is a schematic diagram of a selected pane similar to the selected pane of 500 of FIG. 5 but with extended controls in accordance with an embodiment of the present invention. The selected pane 600 includes a preview pane 602 which contains a preview control 604 having extended controls 614, a metadata display 606 and a tasks display 608. The selected pane 600 also includes a view area 610, in which multiple files or other items 612 may be viewed. The user has selected one of the files 612, so the preview control 604 displays a preview image of the selected file, the metadata display 606 shows properties of the selected file, and the tasks display 608 provides a menu of relevant tasks for operating on the selected file.

The extended controls 614 represent a level of functionality beyond what is typically available from a shell browser. For example, a default preview pane or preview control, such as those shown in FIGS. 2 and 5, may simply display a preview image of a selected item. If the item is a word processing document or slide presentation, the default preview image may be the first page of the document or slide deck. However, by extending the functionality of the preview image to make it more interactive, a user can quite easily manipulate extended controls 614 to page through the document or slide presentation. This enhanced level of functionality improves the user experience because it allows the user to more comprehensively browse the previewed item without opening it, which is particularly useful for files that are not readily identifiable based on the first page alone.

Extended controls 614 can be made available to the user as part of an alternative previewer in a shell browser. The term "previewer" can refer to a preview control or to the a preview pane which includes a preview control. The present invention contemplates a shell browser which provides the user with a default previewer offering a standard level of functionality for multiple item types and one or more alternative previewers offering a different level of

functionality for particular item types to enhance the user experience. Opening up the development of alternative previewers to independent software vendors (ISVs) and other third party developers adds value to the file browsing experience by showing relevant aspects of the file in an easily recognizable way. The present invention contemplates custom previewers for numerous file types and non-file item types including, but not limited to, image files, video files, contacts, games, scanners, video cameras, document files, spreadsheet files, slide presentation files, drawing files and tablet ink files.

The present invention enables a number of scenarios which were not possible with conventional shell browsers, some of which have been described above. Third parties are allowed to describe and demonstrate their file types by providing code that can look inside the file type and provide a meaningful image that a user will understand. For example, Apple could implement a QuickTime™ preview control, which would be displayed when the user selects a QuickTime™ file in the shell browser. This preview control could provide an alternative or extended level of functionality beyond the default previewer in the shell of an operating system, including functionality such as showing the first five seconds of a QuickTime™ movie and/or offering buttons and controls for the user to launch the QuickTime™ player. An alternative previewer for a music file could provide similar extended functionality. As those skilled in the art will appreciate, the possibilities for extended functionality in an alternative previewer are unlimited.

FIG. 7 is a schematic representation of a selected pane similar to FIG. 5 but which also includes a context menu 714 to enable a user to modify metadata in a shell browser in accordance with an embodiment of the present invention. The selected pane 700 includes a preview pane 702 which contains a preview control 704, a metadata display 706 and a task control 708. The selected pane 700 also includes a view area 710, in which multiple files or other items 712 may be viewed. Those skilled in the art will appreciate that, for purposes of the present invention, the metadata control 706 and the task control 708 are not essential features. The present invention contemplates the presence of these and/or other optional features which may assist the user in readily locating a particular item in the shell browser or otherwise enhance the user experience.

The context menu 714 in FIG. 7 presents the user with several options, including the choice of selecting either the default previewer or an alternative previewer for the selected item. The generic text shown in the menu 714 is of course merely one example of the type of options which may be presented to a user for selecting a previewer. A context menu can be provided in any window, including a welcome pane, to improve the user experience. As those skilled in the art will appreciate, any number and variety of context menus could be supported by the present invention. For purposes of the present invention, one means for enabling user selection of a previewer within a shell browser is to provide a context menu such as context menu 714. A user may summon the context menu, for example, by clicking on the corresponding text or object in the preview pane.

Those skilled in the art will appreciate that the present invention contemplates means other than context menus for selecting a previewer for the displayed items from a plurality of available previewers within a shell browser. Another such means is for the user to click on the preview control to enter a selection mode. Similarly, the user may be prompted to select a previewer by right-clicking within the preview pane. By contrast, a user could enter a selection mode by hovering over relevant text or over a relevant object in the preview pane. Numerous alternative means are available and within the scope of the present invention.

FIG. 8A is a flow diagram illustrating a method 800 for enabling a user to select a previewer in a shell browser which supports multiple item types in accordance with an embodiment of the present invention. The method 800 provides a plurality of previewers in the shell browser at 802. The plurality of previewers may include a default previewer for multiple item types and one or more alternative previewers for particular item types. These alternative previewers may include installed applications developed by a third party. At 804, the method 800 presents the user with a choice of two or more previewers for a particular item type. The prompt to select a previewer may be initiated by the shell browser (e.g., upon displaying a new item type) and/or by the user (e.g., by clicking on an object to display a context menu). Upon receiving an input from the user at 806 indicating a selection of one of the previewers for the particular

item type, the method 800 then associates the selected previewer with the particular item type at 808. The selected previewer will remain in use until the user selects a different one. However, if the selected previewer is an installed application, uninstalling the application will also terminate the use of the selected previewer.

FIG. 8B is a flow diagram illustrating a method 810 for automatically selecting a previewer in a shell browser which supports multiple item types in accordance with an embodiment of the present invention. The method 810 provides a plurality of previewers in the shell browser at 812. The plurality of previewers may include a default previewer for multiple item types and one or more alternative previewers for particular item types. These alternative previewers may include installed applications developed by a third party.

At 814, the system (as opposed to the user) automatically and transparently selects a default previewer from two or more available previewers for a particular item type. The system may select a previewer in response to an event such as display of a new item type or the presence of an alternative previewer. The system is configured to select a default previewer based on logical rules. Under exceptional circumstances, the system may decide at 816 to override the rules and select a previewer that would not have been selected under the applicable rules. For example, if the rule is to select a newly available previewer over the current default previewer, an installed application may generally have the authority to change the default previewer to the previewer now available from the installed application. However, the shell browser, for example, may reserve the right to override the change proposed by the newly installed application. For instance, an override may be appropriate when the newly installed application cannot be authenticated as a proper owner of the item type in question.

In any event, the method 810 then associates the selected previewer with the particular item type at 818. The selected previewer will remain in use until a different one is selected. However, if the selected previewer is an installed application, uninstalling the application will also terminate the use of the selected previewer.

Referring next to FIG. 9, a flow diagram illustrates a method 900 for enabling the use of third party previewers in a shell browser which supports multiple item types in accordance with an embodiment of the present invention. The method 900 includes providing a shell browser having a default previewer for the multiple item types at 902. The method 900 further includes providing an extensibility mechanism for third party development of an alternative previewer for at least one of the multiple item types at 904. The alternative previewer may be registered in the shell browser at 906. In the case of an installed application, registration may occur substantially at the time of installation. For example, if the application is installed by an OEM, the alternative previewer may be registered before the user has acquired the computer. Alternatively, the user may install the application locally or remotely.

There are many possible approaches for the extensibility mechanism referenced above in 904. One such approach involves exposing a set of application program interfaces (APIs) so that independent software vendors (ISVs) and other third party developers may develop alternative previewers. With the API approach, a registration mechanism exists which allows an ISV to associate their preview control with an item type owned by the ISV. When an item or file of that type is selected in the shell browser, the ISV's preview control is instantiated via this registration mechanism and the extensibility API. The API provides data to the preview control: data representing the selected item(s) in the view and data representing the parent container of the items in the view. The preview control operates on this data and provides a user interface through the API which is presented in the shell browser. The user may provide input with keystrokes and mouse events which are passed by the shell browser to the preview control which can operate on those user input events.

Those skilled in the art will appreciate that many approaches are possible in the context of the extensibility mechanism of the present invention. In addition to the API approach, similar functionality may be achieved via user configuration, a pointer to HTML or hosting a flash. Moreover, the extensibility model may require that only one application that owns the item type selected may provide only one alternative previewer. In other words, the number of available previewers may be limited to a default previewer and one alternative previewer to

avoid a poor user experience in which multiple registered, extended previewers are in competition with one another. However, another model would be to allow any application that can handle the selected item type to provide one additional previewer. An alternative model would allow any running code to provide one additional previewer for any item type. It may also be desirable under certain circumstances to allow replacement or removal of the default previewer. Many other models are possible and are contemplated by the present invention.

FIG. 10 is a block diagram of a data structure 1000 which is stored on one or more computer-readable media and which contains information indicative of a plurality of previewers in a shell browser. The data structure 1000 includes a default previewer field 1002 containing information indicative of a default previewer which supports multiple item types. An alternative previewer field 1004 contains information indicative of an alternative previewer for a first item type. Another alternative previewer field 1006 may contain information indicative of a second alternative previewer for the first item type, or it may contain information indicative of an alternative previewer for a second item type. Those skilled in the art will appreciate that in some cases there may only be one alternative previewer field, and in other cases there may be two or more alternative previewer fields. The selected previewer field 1008 contains information indicative of whether to invoke the default previewer or an alternative previewer when items of a particular item type are displayed in the shell browser. In the event that field 1006 contains information indicative of an alternative previewer for a second item type, a selected previewer field 1010 may contain information indicative of whether to invoke the default previewer or the alternative previewer when one or more items of the second item type are displayed in the shell browser. The information contained in fields 1002, 1004 and/or 1006 may comprise the previewer code which is configured to run when a user selects an object of that type.

Alternative embodiments and implementations of the present invention will become apparent to those skilled in the art to which it pertains upon review of the specification, including the drawing figures. Accordingly, the scope of the present invention is defined by the appended claims rather than the foregoing description.

CLAIMS

What is claimed is:

1. A shell browser configured to display a plurality of items representing multiple item types, the shell browser comprising: a default previewer which provides a standard level of functionality for multiple item types; and an extensibility mechanism which enables functionality beyond the standard level provided by the default previewer for one or more of the multiple item types.

2. The shell browser of claim 1, wherein the default previewer is a preview control.

3. The shell browser of claim 1, wherein the default previewer is a preview pane.

4. The shell browser of claim 1, wherein the multiple item types include multiple file types.

5. The shell browser of claim 1, further comprising an installed application program which provides the functionality beyond the standard level for at least one item type.

6. The shell browser of claim 1, wherein the extensibility mechanism includes a set of application programming interfaces (APIs) which enable third party software developers to extend the functionality beyond the standard level.

7. An operating system stored on one or more computer-readable media, the operating system including the shell browser of claim 1.

8. A computer comprising a central processing unit, a memory including a shell, and an operating system including the shell browser of claim 1.

9. A shell browser configured to display a plurality of items representing multiple item types, the shell browser comprising: a first previewer which provides a standard level of functionality for the multiple item types; and a second previewer which provides an alternative level of functionality for one or more of the multiple item types; wherein the shell browser is configured to selectively deploy one of the first and second previewers for the one or more item types.

10. The shell browser of claim 9, wherein the first and second previewers are preview controls.

11. The shell browser of claim 9, wherein the first and second previewers are preview panes.

12. The shell browser of claim 9, wherein the multiple item types include multiple file types.

13. The shell browser of claim 9, wherein the shell browser selectively deploys the one previewer in response to an indication from the user.

14. The shell browser of claim 9, wherein the shell browser selectively deploys the one previewer automatically.

15. The shell browser of claim 9, wherein the second previewer is an installed application.

16. The shell browser of claim 9, further comprising a third previewer which provides another alternative level of functionality for one or more of the multiple item types, wherein the shell browser is configured to selectively deploy one of the first, second and third previewers for a particular item type.

17. An operating system stored on one or more computer-readable media, the operating system including the shell browser of claim 9.

18. A computer comprising a central processing unit, a memory, and an operating system including the shell browser of claim 9.

19. A graphical user interface embodied on one or more computer-readable media and executable on a computer, said graphical user interface comprising: a first screen area for displaying a set of items in a shell browser which supports multiple item types; and means for selecting a previewer for the displayed items from a plurality of available previewers.

20. The graphical user interface of claim 19, wherein the plurality of available previewers are preview controls.

21. The graphical user interface of claim 19, wherein the plurality of available previewers are preview panes.

22. The graphical user interface of claim 19, wherein the multiple item types include multiple file types.

23. The graphical user interface of claim 19, wherein at least one of the plurality of available previewers is an installed application.

24. The graphical user interface of claim 19, wherein the means for selecting a previewer is based on a user selection of the previewer.

25. The graphical user interface of claim 19, wherein the means for selecting a previewer is based on an automatic selection of the previewer.

26. The graphical user interface of claim 19, further comprising a second screen area which selects the selected previewer for the displayed items.

27. A computer-implemented method for selecting a previewer in a shell browser which supports multiple item types, the method comprising: providing a plurality of previewers in the shell browser for a particular item type; selecting one of the plurality of previewers for the particular item type; and associating the selected previewer with the particular item type.

28. The computer-implemented method of claim 27, wherein the selecting is performed by the user.

29. The computer-implemented method of claim 27, wherein the selecting is performed by the shell browser.

30. The computer-implemented method of claim 27, wherein at least one of the two or more previewers is an installed application.

31. One or more computer-readable media containing computer-executable instructions for performing the method of claim 27.

32. A computer-implemented method for enabling the use of third party previewers in a shell browser which supports multiple item types, the method comprising: providing a shell browser having a default previewer for the multiple item types; and providing an extensibility mechanism which enables a third party to develop an alternative previewer for at least one of the multiple item types.

33. The computer-implemented method of claim 32, wherein the default and alternative previewers are preview controls.

34. The computer-implemented method of claim 32, wherein the default and alternative previewers are preview panes.

35. The computer-implemented method of claim 32, wherein providing the extensibility mechanism includes exposing application programming interfaces (APIs).

36. The computer-implemented method of claim 32, further comprising registering the alternative previewer in the shell browser.

37. One or more computer-readable media containing computer-executable instructions for performing the method of claim 32.

38. One or more computer-readable media having stored thereon a data structure, said data structure comprising: a first field containing information indicative of a default previewer in a shell browser which supports multiple item types; a second field containing information indicative of an alternative previewer for a first item type; and a third field containing information indicative of whether to invoke the default previewer or the alternative previewer when one or more items of the first item type are displayed in the shell browser.

39. The one or more computer-readable media of claim 38, wherein the default and alternative previewers are preview controls.

40. The one or more computer-readable media of claim 38, wherein the default and alternative previewers are preview panes.

41. The one or more computer-readable media of claim 38, wherein the multiple item types include multiple file types.

42. The one or more computer-readable media of claim 38, wherein the alternative previewer is an installed application.

43. The one or more computer-readable media of claim 38, further comprising a fourth field containing information indicative of a second alternative previewer for the first item type, wherein the third field contains information indicative of whether to invoke the default previewer, the alternative previewer or the second alternative previewer when one or more items of the first item type are displayed in the shell browser.

44. The one or more computer-readable media of claim 38, further comprising a fourth field containing information indicative of an alternative previewer for a second item type and further comprising a fifth field containing information indicative of whether to invoke the default previewer or the alternative previewer when one or more items of the second item type are displayed in the shell browser.

45. The one or more computer-readable media of claim 38, wherein the information contained in the first field is code for the default previewer.

46. The one or more computer-readable media of claim 38, wherein the information contained in the second field is code for the alternative previewer.